INTELLIGENT RESPONSES THROUGH NETWORK-BASED ANSWER DISCOVERY WITH ADVANCED REASONING

Wilson Wong[1], Halizah Basiron[1], Shahrin Sahib[1] and Ong-Sing Goh[2]
[1]Faculty of Information and Communication Technology, Kolej Universiti Teknikal Kebangsaan Malaysia,
Ayer Keroh, 75450, Melaka, Malaysia.
{wilson, halizah, shahrinsahib}@kutkm.edu.my
[2]School of Information Technology, Murdoch University,
6150, Western Australian, Australia.

Abstract
This paper introduces a network-based answer discovery scheme coupled with some advanced reasoning features that is part of NaLURI (Natural Language Understanding and Reasoning for Intelligence), a knowledge-based question answering system. This move beyond classical logic-based reasoning is necessary in order to provide intelligent responses under suboptimal circumstances or failures and is especially important for question answering systems like NaLURI where the nature of the input varies greatly, causing immense uncertainties during response generation.

Key Words
Intelligent response, advance reasoning, answer discovery

1. Introduction

Reasoning deals with facts and beliefs in providing answers to questions about what we know. The reasoning mechanism for discovering answers in many knowledge-based systems has always been based on logic like induction, abduction and deduction. But the problem is that classical logic-based reasoning provides either a correct answer or no answer at all and this is definitely not our definition of what intelligent responses are. To extend beyond such nature of answers, the introduction and formalization some non-logic-based answer discovery approach that complements some more expressive representation language is required. [1] mentioned the need for something like nonmonotonic reasoning, but used this fact as evidence for the inadequacy of logic-based approaches to AI and the need for approaches not based on logic. [2] has even pointed out that the extensions of mathematical logic for reasoning other problems in the logic of AI are beginning to take a definite form including formalization of contexts as objects.

For example, in representation formalism like semantic network where meaning is assigned only by the nature of the programs that manipulate the network, reasoning is only possible via inheritance and intersection search. Even though the appeal of the graphical nature of semantic network has lead to various forms of reasoning that do not fall into standard logical categories, they are all not yet very well documented and formalized. Hence, this paper proposed an alternative form of reasoning for answer discovery which provide intelligent responses under anomalous circumstances.

2. NaLURI Question Answering System

NaLURI [3] is a domain-oriented question answering system designed to scale across multiple domains that attempts full-discourse natural language understanding for both question and information on the World Wide Web. NaLURI features a knowledge-base as its source and employs a novel semantic network reasoning approach to produce direct and justified answers to natural language questions. The knowledge-base is continuously updated with facts extracted from online Cyberlaw news. The ontology and gazetteer will be implemented as domain-dependent modular components, allowing future improvements to achieve openness in domain. We reckoned that the contributions resulted from NaLURI will give way for more researches in the field of question answering to employ higher level of natural language understanding and reasoning, and produce more innovations and improvements. This will indirectly pave way for other intelligent systems capable of understanding natural language and reason with facts. The unpredictable nature and richness of the answers produced by systems based on natural language understanding and reasoning like NaLURI will attract researchers to further embark on studies in the field of intelligent responses.

3. Reasoning in Semantic Network

The basic inference scheme in semantic networks is based on the mechanism of following links between nodes. While it sounds simple, there are two methods to perform the traversal namely intersection search and inheritance. Intersection search works by spreading out from two nodes and finding their intersection to discover relationships among objects. This is achieved by assigning a special tag to each visited node. This method

has many advantages including entity-based organisation and fast parallel implementation. However, very structured questions require highly structured networks. As for inheritance, it is implemented through the *isa* and *instance* representation. Inheritance also provides a means of dealing with default reasoning in semantic networks. For example, we could represent *roosters are birds*, *typically birds fly and have wings* and *roosters run* in the following semantic network depicted in Figure 1.

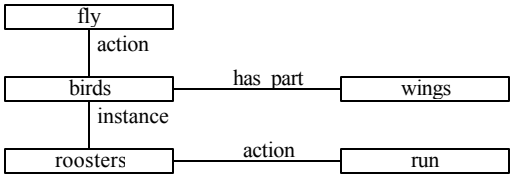

Figure 1: A semantic network for default reasoning

From Figure 1 also, it is obvious that we can say *roosters have wings* even though it is not explicitly stated because the conclusion was drawn through inheritance such that if *roosters are birds* and *birds have wings*, we can immediately deduce that *roosters have wings*. But we obviously cannot come to the conclusion that *roosters fly*. This is when we have to add exceptions such as *roosters do not fly*. Such exceptions will remove or falsified the previous derivation (i.e. *roosters do not fly*).

If a language is such that adding axioms does not affect previous derivations, then it is monotonic. But, in semantic network, inheritance with exceptions is necessary to make it widely applicable and hence, making semantic networks a non-monotonic logic. More generally, any sort of default reasoning is non-monotonic. A very good example of a classical monotonic system is the first-order logic. In making certain inferences, we will also need to differentiate between the link that defines a new entity and holds its value, and the other kind of link that relates two existing entities.

4. XI: A Hybrid Representation Language

XI is a language for representing knowledge about individuals, about classes of individuals and about inclusion relations between classes of individuals. It allows for straightforward definition of multiple inheritance hierarchies and for the association of attribute-value structures with classes or with individuals. XI provides a simple inheritance mechanism which allows attribute values to be inherited by classes or individuals lower in the hierarchy. At one level XI may be viewed simply as a declarative formalism with its own syntax and semantics based on first-order logic. While from another perspective, it may be viewed as yet another logic-based or frame-based inheritance system in the general tradition of KL-ONE [4] or more generally as a form of semantic network [5].

Classes are represented as unary predicates and individuals are atoms. Attributes are binary predicates, the first argument identifying the class or individual of which the attribute holds and the second being the value. XI was chosen as the representation language for NaLURI not only for its simplicity of use, ease of implementation, flexibility and theoretically well-founded, but also for its nature of being a hybrid between frame and semantic network. It allows entities and attributes to be organized around classes, which are later interconnected based on their relationships.

4.1 Two Components of XI

The language has two components, a definitional component and a derivational component. The definitional component allows a hierarchy to be defined and attributes to be associated with nodes in the hierarchy. The definition of a cross-classification hierarchy we refer to as ontology; the definition of a mapping between nodes in the ontology and attributes we refer to as an attribute knowledge base. Together, the ontology and its associated attribute knowledge base form what we term a world model. The derivational component allows one to determine just two sorts of things: whether one node dominates another in the hierarchy and what value an attribute has at a given node. Attribute values are determined first at the given node and then are inherited by working depth-first up the hierarchy from left to right. Multiple values may be obtained by backtracking and nothing is done to prohibit these values from being contradictory, this is left to the application.

4.2 Reasoning using XI

Even though there are non-logical extensions in XI which makes it more powerful, but the reasoning mechanism in XI is still very much constrained by logic-based approaches. Like those of the classical semantic networks, the inference rules in XI are based on inheritance and other hierarchical relationships. There are five types of inference rules which constitute the derivational component of XI in the form of G-clauses:

- Clauses declaring the *a_kind_of* relations between a superclass and a subclass. Let c_1 and c_2 be class terms or variables: $c_1 \Rightarrow c_2$
- Clauses declaring the *is_a* relations between an instance and some class in the form. Let e is an instance symbol or a variable and c is a class term or a variable: $e \leftarrow c$
- Clauses declaring that an attribute holds for an instance. Let e is an instance symbol or a variable and p is an attribute term or a variable:
$$hasprop(e, p)$$
- Conjunctions and disjunctions of the three basic types in the form of G_1, G_2 and $G_1; G_2$.

The inference rule to obtain the type-1 G-clause takes the form of:

$c \Rightarrow D_{1,1}\ \&\ ...D_{1,n1}$ (type-1 O-clause)

$c_1 \Rightarrow D_{2,1}\ \&\ ...D_{2,n2}$ (let c_1 be any class term that exists in some $D_{1,j}$)

$\vdots$

$c_{k-1} \Rightarrow D_{k,1}\ \&\ ...D_{k,nk}$ (let c_{k-1} be any class term that exists in some $D_{k-1,j}$)

$c \Rightarrow d$ (let d be any class term in some $D_{k,j)}$

For example, the let c be *organization(X)* and c_1 be *government(X)* where *government(X)* is a class term that exists in $D_{1,1}$ and because *court(X)* exists in $D_{2,1}$, it can be concluded that class *court* is also a kind of *organization*.

$$organization(X) \Rightarrow government(X) \vee company(X) \vee ngo(X)$$
$$government(X) \Rightarrow court(X)$$
$$\text{---}$$
$$organization(X) \Rightarrow court(X)$$

The second inference rule is used to obtain type-2 G-clause:

$c \Rightarrow d$

$e \leftarrow d_1\ \&\ ...d_m$ (type-2 O-clause where d above satisfies $d = d_i$ for some $1 = i = m$)

$e \leftarrow c$

As a continuation of the example above:

$$government(X) \Rightarrow court(X)$$
$$g1 \leftarrow court(X)$$
$$\text{---}$$
$$g1 \leftarrow government(X)$$

It can be concluded that besides being a court, *g1* can also be considered as a governmental unit.

The third inference rule, used to obtain type-3 G-clause, has two forms:

$props(e,[..., p(e,t),...])$ / $hasprop(e,p(e,t))$

$props(c(X),[...,p(X,t),...])$, $e \leftarrow c(X)$ / $hasprop(e,p(e,t))$

Consider the following example:

$$props(court(X),[name(X,t)])$$
$$c1 \leftarrow court(X)$$
$$\text{---}$$
$$hasprop(c1,name(c1,t))$$

court(X) has the property *names* and because the instance c_1 is a *court(X)*, therefore c_1 also has the property *name* like its class.

The fourth and last type of inference rule sanctioned by XI language involves the conjunction and disjunction of the previous three rules in the form of:

$$\frac{G_1 \quad G_2}{G_1, G_2} \qquad \frac{G_1}{G_1; G_2} \qquad \frac{G_2}{G_1; G_2}$$

5. Advanced Reasoning

The ability of better answer ranking, answer justification, responses to unanticipated questions and resolve situations in which no answer is found in the data sources are a few examples of what constitutes advanced reasoning. Some of the current state-of-the-art in advanced reasoning includes answer explanation, intensional answer description, question relaxation and realization of information fusion [6].

Answer explanation is performed when users have false or unclear understanding of what he or she is asking for [7]. For example, question answering systems using basic reasoning capacity will provide a direct answer *no* to the question *Which bus should I take to reach Kota Kinabalu from Kuala Lumpur?*. With the ability of answer explanation, the answer *there cannot be any bus between Kota Kinabalu and Kuala Lumpur because of the sea* will be produced instead. Reasoning is needed to detect in a question false presuppositions or misconceptions that conflict with the system knowledge base. False presuppositions occur with respect to the database contents while misconceptions usually occur with respect to the database semantics.

Intensional answer description makes generalization or summarization on answers that are too large in scope, making the underlying implications much clearer. For example, the query *Which students have good marks in web programming*? can be answered intelligently in ways like *100% of BITM students, 25% of BITC students*.... To realize this task, reasoning is needed for cleaning answers when they partly overlap and for determining whether an answer is more specific than another one and for organizing mutually consistent answers.

In question relaxation [8], neighborhood information corresponding to the question is used while reasoning for the answer. For example, the question *What are the Chinese restaurants available in Penang?* would have a small set of solutions and will be more informative if the scope is extended to provide information on hawker centers and restaurants of other kinds. One way to increase the yield of potential answers is to find within the ontology, a set of most appropriate concepts which are conceptually close to the relaxed concept in the initial question. Three types of relaxation techniques are

available namely rewriting predicate, broadening of the domain of a variable and breaking a join dependency.

6. Answer Discovery in NaLURI

The reasoning mechanism of the NaLURI question answering system couples the novel idea of complexity reduction during answer discovery in a network-oriented knowledge base with two advanced reasoning features namely relaxation of event constraint and explanation on failure to provide higher standards of responses, way beyond the current conventional factual answers. Such reasoning mechanisms and the ontological commitment of the knowledge base can be said as a match made in heaven. This statement is well-justified because such advanced reasoning cannot be carried out without the use of domain ontology and knowledge base and only with the adoption of these high-level reasoning capabilities can the ontological information and knowledge base be thoroughly exploited.

In the first two subsections, we will discuss how the complexity of discovering for answers in network-oriented knowledge base can be handled in terms of complexity reducibility through a series of form reduction. We then introduce the algorithmic ideas for advanced reasoning capabilities that are implemented as part of the answer discovery mechanism in the third section.

6.1 Query Network

Reasoning in NaLURI works on two types of logical network namely the semantic network and query network, consisting of the networked representation of the meaning of online news and of the question respectively. In essence, the query network is similar to semantic network as appeared in the knowledge base and the derivation of both networks uses the same implementation of natural language understanding. The only difference during the construction of a query network is the use of potential answer marker X. The marker can appear only in leaf nodes in the query network or in other words, only attributes of entity objects can be returned as answers. The placement of the marker is carried out during the discourse integration phase of the natural language understanding process. The placement can either occur naturally on subjects and objects of verbs or explicitly placed for noun triggered events and in the absence of any marker. In the first case, subjects and objects of a verb can be missing in a dependency tree of a question. Consider the example of two simple questions, *Who sues Microsoft?* and *Microsoft sues whom?*. The corresponding dependency trees are shown in Figure 2.

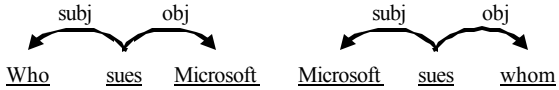

Figure 2: Dependency structure of questions

The verb *sue* would give rise to a *legal_proceeding* event and its associated pattern and map as shown below in Table 1. As the wh-word assumes the position of a subject in the first question and object in the second question, the answer marker is placed in the attributes of the event that are mapped to those positions namely *plaintiff* in the former and *defendant* in the latter.

Name	Category	Pattern	Map
sue	legal_proceeding	{LEGAL_ENTITY} <RELATION> {LEGAL_ENTITY}	{PLAINTIFF} <RELATION> {DEFENDANT}
file on	filing	{VARIABLE} <RELATION> {DATE}	{} <RELATION> {OCCUR_ON}
filing	filing	no pattern	no map

Table 1: Sample gazetteer entries for *sue*, *file on* and *filing*

In the next case where events are triggered by nouns or as a contingency step in the absence of any markers, markers are placed using the help of wh-words. Consider the question *When was the filing of the case against Excite by Microsoft?*, where the noun *filing* would give rise to the *filing* event. Given the event category *filing* and the entity category *date* indicated by the wh-word *when*, NaLURI search for trigger words in gazetteer that belong to the event category and consist of the entity category in its pattern. Each trigger word in the gazetteer has an associated map for assigning the matching values to their appropriate attributes and using this facility, the corresponding attribute in the map for the matching entity category is assigned with the answer marker X as its value. By referring to the sample gazetteer entries above, the trigger word *file on* satisfies the requirement of being in the same category as the noun *filing* and having *date* in its pattern is *file on*. Given that, the corresponding map for the sub pattern *date*, which is the attribute *occur_on* will be assigned the answer marker X as its value.

The network shown below in Figure 3 is a complete example of the query network for the question *When did AT&T file its case against Microsoft?*. The understanding module has placed an answer marker X in the date entity *a039* using the contingency step. Let's analyze the question in steps. Firstly, the verb *file* has the subject *AT&T* and the object *its case*. Based on the sample entries below in Table 2, the verb gives rise to the event *filing* and both the subject and object satisfy the pattern, resulting in entity object *AT&T* being assigned to the event attribute *plaintiff*. This *plaintiff* relation is represented as an edge between nodes *6360* and *1b1c0*. As for the preposition *against*, it triggers the general event *legal_proceeding* and its direct object *Microsoft* is assumed as value for the event attribute *defendant*, represented as an edge connecting nodes *bf99* and *1b1c0*.

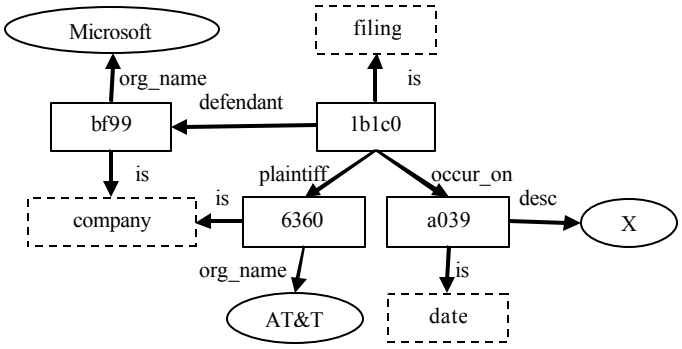

Figure 3: Query network for *When did AT&T file its case against Microsoft?*

At this stage, no more grammatical relationships can be exploited and the answer marker has not been assigned. The contingency step is executed whereby the gazetteer is explored to find trigger words belonging to the same category as the previous events namely *filing* and *legal_proceeding* and having the entity category for the wh-word *when*, which is *date*, in its pattern. Based on the sample gazetteer entries below in Table 2, the trigger word *file on* for event *filing* and *occur on* for event *legal_proceeding* are found. Consequently, the answer marker X is assigned to the map, *occur_on* of the corresponding sub-pattern *date* for each trigger word.

Name	Category	Pattern	Map
file	filing	{LEGAL_ENTITY} <RELATION> {VARIABLE}	{PLAINTIFF} <RELATION> {}
against	legal_proceeding	{FILING|VARIABLE|LEGAL_ENTITY} <RELATION> {LEGAL_ENTITY}	{}<RELATION> {DEFENDANT}
file on	filing	{VARIABLE} <RELATION> {DATE}	{}<RELATION> {OCCUR_ON}
occur on	legal_proceeding	{VARIABLE} <RELATION> {DATE}	{}<RELATION> {OCCUR_ON}

Table 2: Sample gazetteer entries for *file*, *against*, *file on* and *occur on*

6.2 Answer Discovery using Selective Network Path Matching

After obtaining the query network, the task of answering the question is reduced to discovering the presence of the query network in the whole of semantic network. To perform the discovery, let's first understand the notion of the three types of nodes is a query or semantic network. A root node is a class node where one or more intermediate nodes are being created and thus, in the network, a root node can only have incoming edges, usually more than one. The second type of node is intermediate node where edges can be from both directions to interconnect other intermediate nodes. The last type of node is the leaf node. The leaf node is similar to root node in the sense that both can only have incoming edges and beyond that, the similarity ends. In the leaf node, there can only be one incoming edge. To perform the discovery, query and semantic network are collapsed into two sets of all possible paths Q and S respectively. Each set of paths takes the form of $\{P_1,...,P_n\}$ where P_i is a sequence of alternating node and edge in the form $n_{i1}, e_{i1}, n_{i2}, e_{i2}, n_{i3}, e_{i3}, n_{i4}$ that satisfy the rules:

- each path sequence must begin with a leaf node and ends with a root node;
- each path sequence must contain exactly two intermediate nodes; and
- for query network, the path beginning with the leaf node X must not be included in Q.

Another set called A, consisting of only one element, which is the path in the query network that begins with the leaf node X is introduced. Hence, it can be concluded that the answer may be present and retrievable from the semantic network if for each $q_i \in Q$, there exists an $s_i \in S$ such that q_i conditionally matches s_i. The condition of the matching between elements of Q and S implies a literal match between member n_{i1}, e_{i1}, e_{i2}, e_{i3} and n_{i4} of path sequence q_i and of s_i.

Then, the problem of discovering the answer has been reduced to finding the path in S that matches every appearance of nodes and edges except n_{i1}, e_{i1}, n_{i2} and n_{i3} in the sole element of A. Consider the sample query network in Figure 3 for the question *When did AT&T filed its case against Microsoft*. The entire network is collapsed into the set Q below:

Q = {"Microsoft, org_name, bf99, defendant, 1b1c0, is, filing", "AT&T, org_name, 6360, plaintiff, 1b1c0, is, filing"}

A = {"X, desc, a039, occur_on, 1b1c0, is, filing"}

Based on a segment from the knowledge base in Figure 4, the set S is obtained:

S = {"Microsoft, org_name, bf99, defendant, 1b1c0, is, filing", "AT&T, org_name, 6360, plaintiff, 1b1c0, is, filing", "2002, year, a039, occur_on, 1b1c0, is, filing", "federal court, org_name, b7, occur_at, 1b1c0, is, filing", "federal, court_type, b7, occur_at, 1b1c0, is, filing"}

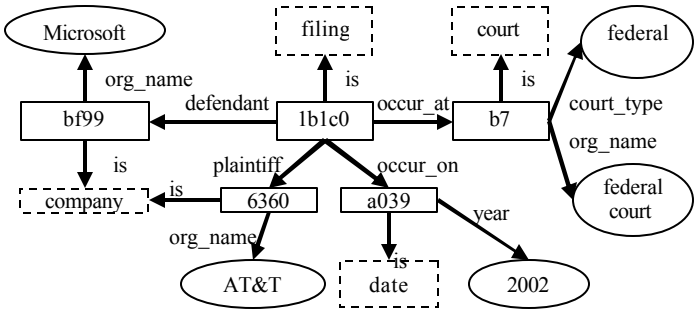

Figure 4: A portion of the semantic network from knowledge base

By selectively matching the members of path sequence q_i and s_i, all paths in Q are satisfied and the next step would have the sole element of A matched against elements of S, to have the leaf node X of the path sequence in A assigned with the value of the corresponding member n_{ml} of the matching S_m. In our case, X would be assigned to *2002*.

6.3 Advanced Reasoning Features in NaLURI

The two algorithmic ideas implemented during answer discovery to provide better answers over conventional techniques are discussed in this last section of the reasoning mechanism. Firstly, relaxation of the event constraint is done to enable the refocusing of the question scope for questions extending beyond a single event to return answers of related events.

Consider the example question *Who presided the case against Microsoft* where it would produce the path set *Q = { "Microsoft, org_name, bf99, defendant, 1b1cc, is, legal_proceeding" }*

The root node *legal_proceeding* at the end of the sequence is actually a superclass node that is capable of having many others subclasses namely *filing*, *resolution*, etc. Because the focus of the question is wide, at the level of superclass event, the chances of finding the desired answer are very slim as meanings of news are stored in their most detailed level in the semantic network to leave no room for ambiguities. Knowing this, the literal matching of n_{i4} during the selective path matching is extended so that parent-child relationship is put into consideration. During the matching of $q_i \in Q$ and $s_i \in S$, the member n_{i4} of path q_i is relaxed to enable the subclasses of n_{i4} and in our case *legal_proceeding*, to pass through. This is to say that any of the below sample path s_i can have a positive match with q_i.

s_1 = Microsoft, org_name, bf99, defendant, b2, is, resolution
s_2 = Microsoft, org_name, bf99, defendant, b2, is, appeal
s_3 = Microsoft, org_name, bf99, defendant, b2, is, filing

Secondly, explanations on failures are provided in the absence of any valid answers to clear out any doubts concerning the status of the knowledge base. Consider the example question *Which judge presided the ruling of the case by RealNetworks against Microsoft*. A conventional question answering system would easily answer *No* in the absence of any valid answers. Such answers will definitely leave the users with a big question mark concerning the actual implication. The answer *No* can have two implications here. First, the system said no because of the inexistence of such case between RealNetworks and Microsoft and second, the system can also say no because of the absence of any information about the judge presiding the closing of the case in its knowledge base.

To provide an explanation of why no valid answers were produced, the following conditions are used:

- if there exist at least one q_i in Q that fails to match some s_i in S, then the response stating that *no valid answers can be discovered because the event in question does not exist* will be used; and
- if all q_i in Q holds for some s_i in S and the sole element in A fails to find a match in S, then the response stating that *no valid answers can be discovered even though the event in question exists because of inadequate knowledge about the event* will be used.

7. Conclusions

This paper has highlighted the inadequacy of the default reasoning or inference mechanism in conventional knowledge-based systems due to the constraint of logic. This has resulted to the underutilization of many powerful representation formalisms like the semantic network underutilized during answer discovery in question answering systems. Hence, we have proposed a network-based answer discovery approach that practices selective path matching and complexity reduction for manipulating the semantic network when searching for answers. The approach is also combined with the power of advanced reasoning features to cater various anomalous situations and to generate dynamic responses. This approach will act as an alternative for researchers who aim to develop knowledge-based question answering systems that have the capability of synthesizing dynamic responses.

References:

[1] Minsky, M. (1975). A Framework for Representing Knowledge. In *The Psychology of Computer Vision*. McGraw-Hill.

[2] Dreyfus, H. (1992). *What Computers Still Can't Do*. MIT Press.

[3] Wong, W. (2004). *Practical Approach to Knowledge-based Question Answering with Natural Language Understanding and Advanced Reasoning*. Thesis (MSc), Kolej Universiti Teknikal Kebangsaan Malaysia.

[4] Brachman, R. & Schmolze, J. (1985). An Overview of the KL-ONE Knowledge Representation System. *Cognitive Science 9*(2):171-216.

[5] Sowa, J. (1991). *Principles of Semantic Networks: Explorations in the Representation of Knowledge*. Morgan Kaufmann.

[6] Gaasterland, T., Godfrey, P. & Minker, J. (1992). An Overview of Cooperative Answering. *Journal of Intelligent Information Systems, 1*(2):123-157.

[7] Benamara, F. & Saint-Dizier, P. (2003). Dynamic Generation of Cooperative Natural Language Responses. In *Proceedings of the EACL Workshop on Natural Language Generation.*

[8] Benamara, F. & Saint-Dizier, P. (2004). Advanced Relaxation for Cooperative Question Answering. In *New Directions in Question Answering*. MIT Press